Wavefront Control Testbed Integrated Software System

Laura A. Burns[*a], Scott A. Basinger[†b], Terrence L. Beck[‡c], Jennifer E. Deering[§d], Daoanh Tonnu[**e,]
Don J. Lindler[††c], Andrew E. Lowman[‡‡b], Robert O. Morris[§§b], Catherine M. Ohara[***b],
Peter P. Petrone III[†††f], David C. Redding[‡‡‡b], Jeffrey C. Schott[§§§g], Sara M. Stoner[****a],
J. Ladd Wheeler[††††h],

[a]Science Systems and Applications Inc.;
[b]Jet Propulsion Laboratory, California Institute of Technology;
[c]Sigma Space Corporation;
[d]General Dynamics Decision Systems;
[e]Jackson and Tull;
[f]ManTech Systems Engineering Corporation;
[g]Scientific Data Systems;
[h]The Hammers Company, Inc.

ABSTRACT

The Wavefront Control Testbed (WCT) is used to demonstrate the wavefront sensing and control algorithms and procedures that will be used on the Next Generation Space Telescope (NGST). The Segmented Telescope Control Software, written in MATLAB®, is the primary development and operational tool used. The software has an extensive graphical user interface that allows the user to interact with the hardware and algorithms.

A variety of additional software programs support the Segmented Telescope Control Software (STCS). Various hardware control software interacts with MATLAB via TCP/IP connections. When access to the hardware is unnecessary or undesirable, we can access the model server that simulates the system. A stand-alone safety monitoring LabVIEW® program alerts technicians if a hardware failure occurs. A C program gives the operator a graphical way of monitoring the network connections to the various systems. An Interactive Data Language® (IDL) data archiving routine creates a database to monitor and maintain the testbed data and executes the MATLAB to Flexible Image Transport System (FITS) translator. Additionally we have implemented a web-based bug tracking and plan to add experiment scheduling and a document archive.

Due to the nature of the testbed, these software programs are constantly evolving, causing a variety of challenges over the years. This poster will describe these software elements and the issues that have arisen trying to use them together.

Keywords: NGST, WCT, DCATT, Segmented Telescope Control Software

---

[*] Laura.A.Burns@gsfc.nasa.gov; phone 301-286-9830; fax 301-286-7021; NASA Goddard Space Flight Center, Mail Stop 443, Greenbelt, MD, USA, 20771-0001; NGST's WCT site: http://www.ngst.nasa.gov/Hardware/text/WCT.html
[†] Scott.A.Basinger@jpl.nasa.gov; phone 818-354-3065; Jet Propulsion Laboratory, California Institute of Technology, M/S 306-451, 4800 Oak Grove Drive, Pasadena, CA, USA, 91109-8099;
[‡] Terry.Beck@gsfc.nasa.gov
[§] Jennifer.Deering@verizon.net
[**] atonnu@jnt.com
[††] lindler@rockfit.gsfc.nasa.gov
[‡‡] Andrew.Lowman@jpl.nasa.gov
[§§] Robert.O.Morris@jpl.nasa.gov
[***] Catherine.Ohara@jpl.nasa.gov
[†††] Peter.Petrone@gsfc.nasa.gov
[‡‡‡] David.Redding@jpl.nasa.gov
[§§§] schott@tharsis.gsfc.nasa.gov
[****] sstoner@hst.nasa.gov
[††††] lwheeler@hammers.com

1. INTRODUCTION

When discussing the Next Generation Space Telescope (NGST) Wavefront Control Testbed (WCT) we usually refer to the Segmented Telescope Control Software (STCS)[1], the hardware[2,3], the experimentation[4] or the algorithms[3,5,6,7]. Until now, we have neglected to discuss the layers of software and hardware that enable the STCS to interact with the hardware. Without these layers, the remote communication capabilities, so pivotal to the operation of the testbed, would be impossible.

The WCT is a large testbed with small optics used to test the wavefront sensing and control algorithms that may be used on NGST. The hardware is located at NASA's Goddard Space Flight Center (GSFC), but can be controlled via the Internet from the Jet Propulsion Laboratory (JPL) or other remote locations. The testbed contains two deformable mirrors, three small spherical mirrors each controlled by six actuators, a Charged Coupled Device (CCD) camera, a quad cell detector, a white light source, a 633 nm laser, various filter wheels, and various linear and rotation stages[2]. When the user wishes to run an experiment with this hardware, s/he runs the STCS, written in The MathWorks MATLAB® with an extensive Graphical User Interface (GUI)[1]. The user then connects to the hardware via the laboratory computing environment, discussed in Section 2. The user may also test procedures or software modifications by connecting to the model server, discussed in Section 3, which uses the same command formatting to simulate the hardware. Section 4 discusses how these commands are formatted and sent. In Section 5, we discuss how the STCS takes advantage of distributed computing for computationally intensive data processing. Section 6 describes the data archiving procedures. Finally, in Section 7 we discuss the bug tracking system used by the STCS.

2. LABORATORY COMPUTING ENVIRONMENT

Figure 1 graphically depicts the laboratory computing environment. In each of the subsequent sections, we will describe various pieces of this system. The remote operator can be physically located anywhere geographically, as long as there is a network connection and the software is appropriately installed. We regularly operate the software from both Maryland and California.

Within the past year, GSFC has implemented new firewall rule set concerning connection activity. If a connection is established, and the appropriate commands are not transmitted within the appropriate time limit, the firewall will kill the connection. This failure is referred to as the short-term time out. Additionally, if a connection is established correctly, but remains inactive for more than a specified time interval, the firewall will kill the connection. This is the long-term time out. The WCT team was not informed when these rules were implemented, and therefore spent several months attempting to diagnose the problems we had been seeing with the software. When one of these time out instances occurs, the two connected programs can't detect the broken connection. In some cases, the only way to force the software to release the connection port is to restart the software on the control computers. The short-term time out rule was easy to work around, but the long-term time out rule has proven to be more difficult. We will mention how each rule has affected each of the various systems below.

2.1. Internal Network

The gateway to the control software and hardware is the Gateway Sun workstation. This machine acts as a router for the rest of the control computers in the laboratory. This was done in order to improve security for the system. With this setup, we can allow access to the hardware from an outside system for short periods of time, and then block all access after the designated time period has elapsed.

For data processing and archival purposes, the Gateway and Data Manager Sun workstations share their hard drives using Network File System (NFS), allowing files to be shared between the two computers. Due to required office moves, we have had to move the Data Manager to a separate network at GSFC. Because of the security precautions in place on each of the two networks, we have had problems with the NFS mounting and maintaining constant communications between the two machines. The long term timeout mentioned above appears to be adding to the problems. Even though both computers are physically located at GSFC, the fact that they are on different networks forces the connection to go through the firewall. Periodically the NFS connection is broken by the firewall, and the Gateway computer must be restarted in order to reestablish the connection. We attempted to implement "keep alive" procedures, but were unsuccessful in preventing the failures. We are now looking for ways to move the Data Manager computer back onto the original network.

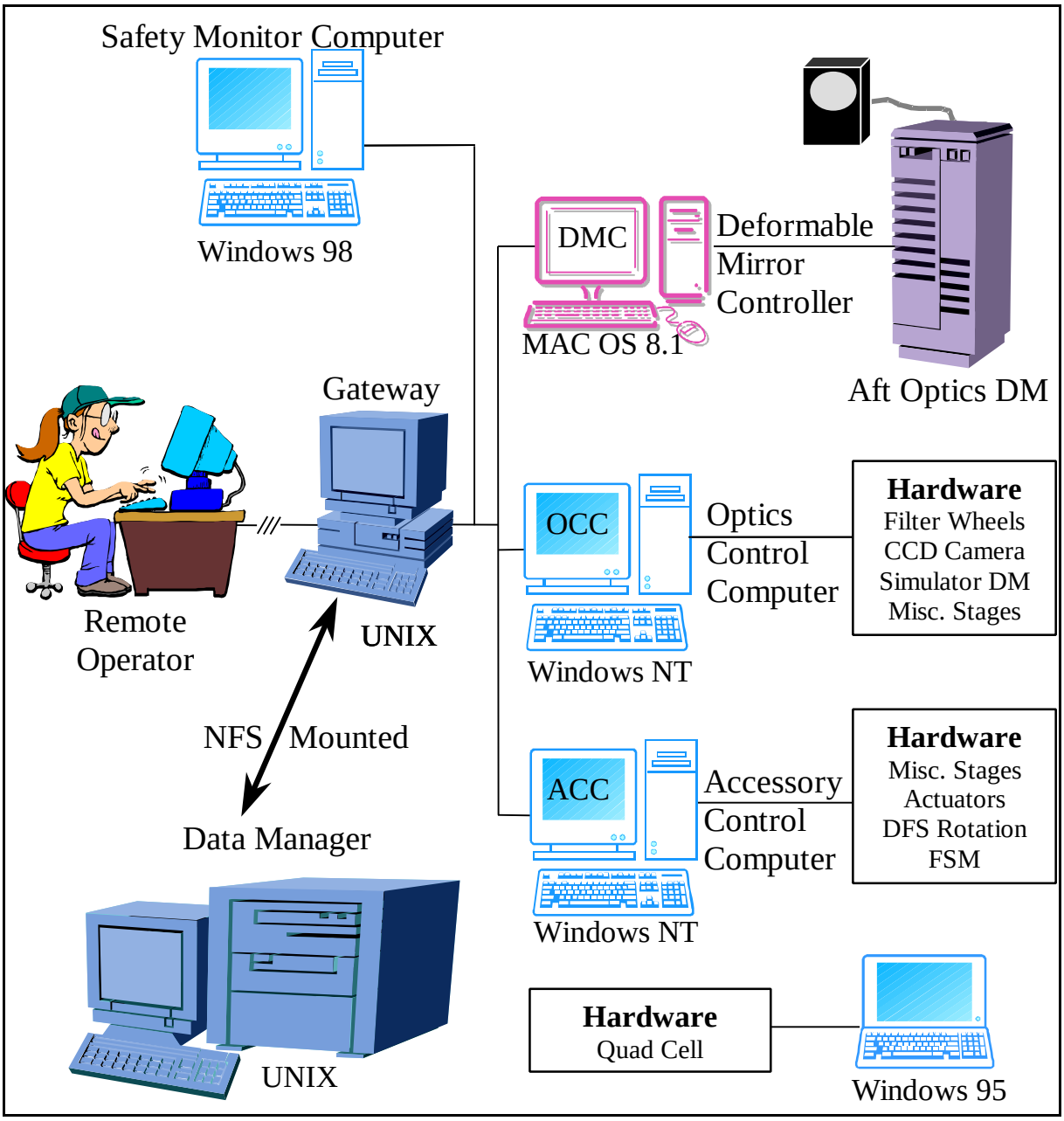

Figure 1: Laboratory Computer Environment

2.2. Optics Control Computer (OCC)

The first of the control computers is the Optics Control Computer or OCC. The term OCC is used for both the physical computer itself, and the software that runs on that computer. This Windows NT® Personal Computer (PC) is dedicated to running this one piece of software, so the dual use of the acronym is reasonable. The OCC software was written using National Instruments LabWindows®, thus the graphical interface code was easy to generate, and the body of the code is familiar to anyone who knows the C programming language. The hardware can be controlled directly from the PC using the GUI, or remotely using the command formatting described below.

The computer/software combination controls the power to the white light source, the four linear stages, six filter wheels, a rotation stage, the external shutter, the CCD camera and the Simulator Deformable Mirror (SMDM)[8]. The computer is connected to the hardware via two RS232 ports, a PCI frame grabber card, a National Instruments Data Acquisition (DAQ) card, and Newport motion controller card.

The firewall problems mentioned above were not initially evident for this computer/software combination. Since the OCC controls the CCD camera and the camera translation stage, the majority of the hardware commands are sent through this computer. Once the nature of the problem was discovered, we were able to demonstrate the behavior on this machine as well. "Keep alive" commands have been implemented in the socket layer to maintain the connection for long time periods.

A modified version of the OCC software has been utilized on the Phase Retrieval Camera (PRC)[9,10]. While the baseline code and the TCP/IP command formatting has remained relatively the same, the PRC-OCC differs in that it has been separated into two independent programs: a "server" application consisting of the basic hardware control functions with a minimal GUI, and a "client" application consisting of the full-scale, user-friendly GUI, which communicates with the server using the same communication protocol as the STCS[8]. This allows the user to control the hardware from a remote location via the PRC-OCC client software. The PRC-OCC is currently the only control computer for the PRC. [9,10]

2.3. Deformable Mirror Control (DMC) Computer

Of all of the control computers, the Deformable Mirror Control (DMC) computer is the only one dedicated to running only one device, the Aft-Optics Deformable Mirror (AODM). The AODM is a six inch Xinetics 349 actuator deformable mirror[11]. Due to the nature of the provided drivers, the DMC was developed on a Macintosh® running OS 8.1. We have been unwilling to upgrade the operating system due to possible software instabilities that may result. The DMC software was written using Code Warrior Professional®, Version 3, producing C/C++ code that makes extensive use of MetroWerks'® Power Plant classes[11]. The software has a detailed GUI, allowing both direct and remote control.

The DMC computer has been susceptible to the firewall timeouts as well, primarily the long-term time out. Additionally, due to the existing team's unfamiliarity with Macintosh systems, we have yet to discover the appropriate solution, but we expect to soon. The use of the AODM varies from experiment to experiment, but in many instances we are able to work around the problem by applying the initial figure to the AODM, and then disconnecting from the computer for the remainder of the experiment.

2.4. Accessory Control Computer (ACC)

The Accessory Control Computer (ACC) is dedicated to running the ACC software, and thus the acronym ACC is often applied to both the hardware and the software. The user can control the hardware remotely or via the GUI interface on the Windows NT PC. Currently, the ACC controls a rotation stage, a translation stage, the Fast Steering Mirror (FSM), and the actuators for the WCT-2 small optics. The WCT-2 small optics consists of three small, round, spherical mirrors. A set of three New Focus picomotor actuators and three Physik Instrument PZT actuators provide tip, tilt, and piston control for each mirror[2]. The ACC connects to the stages via a Newport controller card and to the actuators via a series of daisy-chained GPIB interfaces.

Usually, FSMs are used to remove jitter and drift from a system in order to improve image stability. In the case of WCT, the line of sight drift and jitter environment are acceptable for standard operation[2]. In this case, the FSM will be used to induce jitter and drift into the system to test the limits of the wavefront sensing and control algorithms. The ACC communicates with the FSM via an analog to digital (A/D) 8-channel PowerDaq board to simulate any desired jitter or drift profiles. The current signal generation capability is to generate, in real time, a waveform based arbitrary combination of sine waves, square waves, and triangle waves with varying amplitude, frequency, phase, and timer responses computed from specified digital finite impulse response (FIR) or infinite impulse response (IIR) transfer functions. As the testbed grows, we may use the FSM in combination with a quad-cell controller or CCD camera to allow closed-loop control to remove what line of site jitter exists in the system.

This computer was the most susceptible to the firewall connection errors. The connection sequence sent by the Executive was not sufficient to prevent the short-term failure, and the hardware was often not commanded enough to

avoid the long-term failure. Initially, since the problems were so evident on this computer, we focused on the ACC software itself. Some modifications were made and tested, but failed to improve the problem. The modifications did shed light on the nature of the trouble and helped us redirect our efforts to the firewall. The "keep alive" solution used on the OCC, was first implemented, tested, and verified on the ACC.

2.5. Monitor Software

The Monitor software runs on the Gateway computer and provides the user with basic information about the status of the system. This software controls no hardware, but provides the user with a way to monitor the health of the system. Although the Monitor runs on the Gateway, it can be viewed on remote computers via X-Windows display software. There is nothing to control in this software, and it can't be connected to the STCS.

The various control programs connect to the monitor server and every time a message is passed between the control software and the STCS, the command is mirrored on the Monitor. Additionally, the monitor displays the network commands as well as several status lights showing the health of the system. As the status of the various controllers change, so do the colored blocks. The possible colors are green, yellow, red, white and black. Black indicates that the software is not currently connected to the monitor, and therefore the monitor has no information on the status of the system. When the indicator is white, the control software has connected to the monitor, everything is nominal, and there is no STCS connection to that system. A green status is similar to white, except that the STCS has successfully connected to the control software. Yellow indicates that they system can be controlled either remotely or locally, but otherwise, the system is normal. Red indicates a hardware error. [16,17]

In Figure 2, the OCC block is displayed as red since there was a CCD initialization error. The Optical Telescope Assembly (OTA) Controller block is black indicating that the software has not connected to the Monitor. This connection block was added when we first planned to have a large primary mirror. The original mirror was never built, so we have not needed an OTA Controller, but never bothered to remove the related block. The Deformable Mirror (DM) Controller is running optimally, so the status is green. The yellow Accessory Controller, ACC, block designates that it is in a state where the software can be controlled either directly or remotely.[16,17]

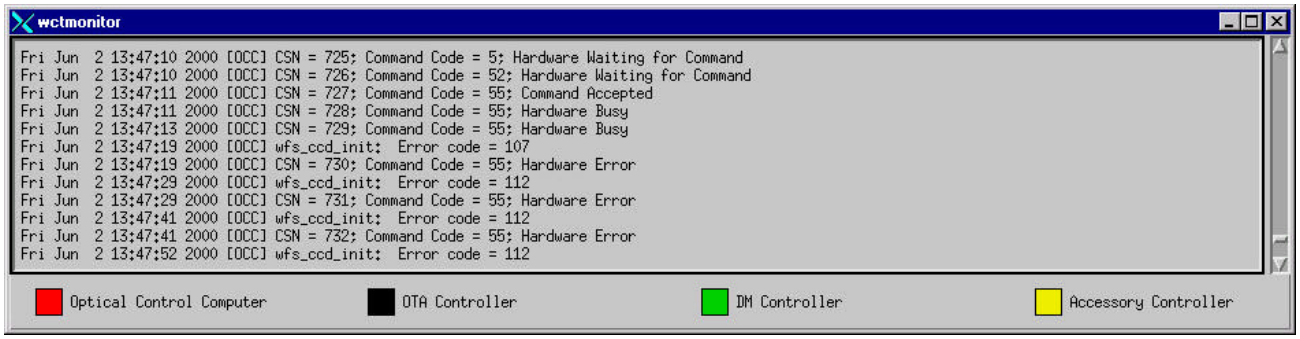

Figure 2: WCT Monitor Screen Shot

2.6. Safety Monitor Computer

The safety monitor computer and software do not directly control any hardware. The software, running on a Windows 98® PC, can not be remotely controlled. The safety monitor is part of the internal laboratory network, but is unable to connect to the STCS or Monitor Software. The software is written in LabVIEW®, and monitors the CCD camera chiller flow rate, the lamp chiller flow rate, the AODM power levels, the SDM power levels, airflow, and temperature. If an anomaly occurs, the safety system will contact the testbed team via e-mail and pagers.

The safety monitor computer is connected to a SCXI National Instruments chassis via a National Instruments PCI card. The chassis contains various modules that measure the health of the specified hardware. A series of Pulizzi relay boxes, each controlling the power to one or several of the aforementioned devices, are connected to the computer via a National Instruments serial port expansion card. If the monitored device goes out of range, action is taken. The one exception is the temperature: if one of the 17 temperature sensors is above the specified level, no action is taken. If the air flow drops

below the specified limit, the safety system will alert the team, but take no further action. If a problem occurs with any of the other hardware elements, the safety software will flip the relay, killing the power for that device and several associated devices. Then the system will alert the team. Manual intervention is required to restore the laboratory to operational conditions.[18]

Since this system is focused on the internal laboratory environment, we have not had the networking difficulties evident with the control computers. Currently, the safety monitor software does not communicate with the STCS. If an error occurs during experimentation, the user will not be informed directly through the STCS. A long term goal for this system is to develop a way to interface the control software with the STCS to allow real time monitoring of the hardware.

2.7. Quad Cell Controller

Data from the quad cell is collected using a dedicated laptop computer running Windows 95®. The output from the quad cell is delivered to an analog signal processor and then to an analog-to-digital (A/D) converter card installed in the laptop computer. The signal processor outputs a voltage in X and Y that is proportional to the image displacement at the quad cell. These signals are sent to the A/D converter and saved on the laptop computer. The position of the quad cell is controlled by three stages allowing x, y, and z translations. These stages are controlled via a parallel port to GPIB converter.

The network communication problems mentioned above do not affect this system since the laptop is not connected to any network. Any information acquired using the quad cell must be manually transferred to another system.

In the past, this system has only been used for specific jitter tests, but in the future the usage of the system may expand. If this is the case, the software will need to be re-written in order for it to be interfaced with the standard software system.

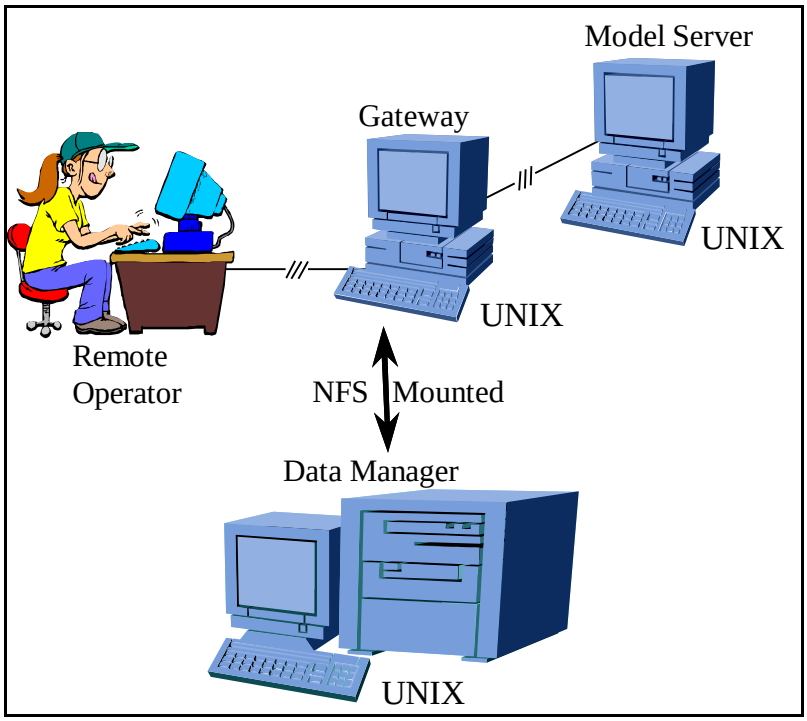

Figure 3: Model Server Computer Configuration

3. MODEL SERVER

The model server is software that resides on a UNIX workstation and is used for training, testing, and validation. As shown in Figure 3, the STCS software interfaces to the testbed in the same method as the hardware. It uses the same commands and command formatting as would be sent to the hardware, but sends them to the server process instead.

The code is written predominantly in FORTRAN, with some communication routines written in C. The code interfaces with the Modeling and Analysis for Controlled Optical Systems (MACOS) models creating simulated images for use in testing and analysis. For more information on the model server and MACOS, see Reference 1.

4. COMMUNICATION WITH THE STCS

4.1. The Internet Connection

The remote computer talks to the control computers through the Internet. The Gateway computer acts as a firewall and isolates the control computers from the rest of the Internet. The control computers are accessed via Network Address Translation (NAT), which is implemented on the Gateway computer. Each control computer has its own IP address to which NAT allows only certain computers to connect. The actual communications protocol is Berkeley Sockets. Each control computer has a specific port that is connected to via a Berkeley socket and binary commands are sent back and forth between the remote computer and the control computers.

4.2. Command Syntax

The binary commands sent to the control computers were derived specifically for the hardware on the testbeds. The commands are rigidly defined in Interface Control Documents and all numerical values sent must be "little-endian" to conform to PC standards. Each command consists of several fields that 1) identify the control computer the command is meant for, 2) define a sequence number, 3) specify which piece of hardware is being accessed, and 4) give the value to set the hardware to. In addition, the position of a particular filter wheel or translation stage may be queried. After receiving the command, the control computer sends back a response that the command has been received and returns either a message that the command was successful or returns an error message if there was a problem commanding the hardware. This methodology has worked well for us and even allows the user to run from a laptop connected through an analog modem.

For more information on the communication with the STCS, see Reference 1.

5. DATA PROCESSING

The phase retrieval algorithm of the STCS requires significant computing power. The distribution of this processing load is handled differently, depending on the location of the remote operator. The differences between JPL and GSFC are described below.

5.1. GSFC

The computing environment at GSFC is shown in Figure 1 and Figure 3. The key processing aspects of the system are the Gateway and Data Manager UNIX machines. The Gateway has two 200 MHz UltraSPARC CPUs and the Data Manager has four 400 MHz UltraSPARC II CPUs. The STCS can distribute the phase retrieval processes between the two computers to optimize efficiency. A standard phase retrieval process takes approximately four minutes to process on the Data Manager computer.

5.2. JPL

The environment at JPL, shown in Figure 4, varies from that of GSFC in a couple of fundamental ways. The user has the choice of sending the process to a beowolf or a network of Sun workstations. The Beowulf is created out of four to five PC's running Linux. The Intel Pentium III processors range from 866 MHz to 1 GHz. If using the Beowulf, a standard phase retrieval takes approximately two minutes.

The network of Sun workstations at JPL has a variety of processors and computers. The STCS software allows the user to intelligently distribute the processes to computers with lighter loads. The processing time for this system is similar to that of the GSFC system, about four minutes, but that number will vary depending on the computing load of the network and the speed of the CPUs.

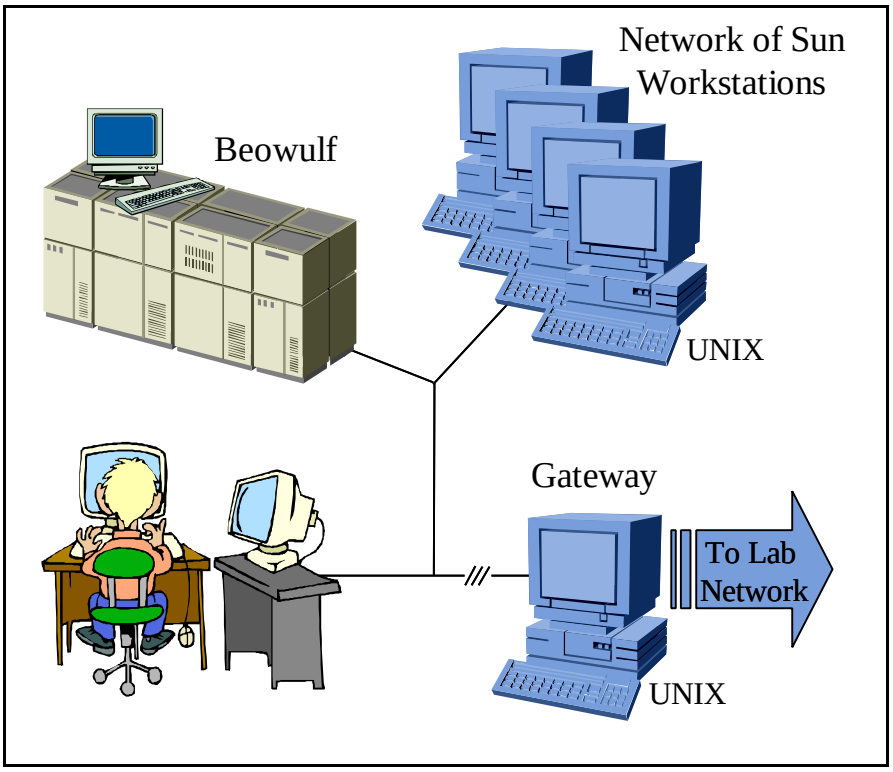

Figure 4: JPL Processing

6. DATA ARCHIVING

We implemented a relatively simple system for data archiving. The data is stored on the Data Manager computer in a specified directory structure. Within this structure, a directory name standard is used to define names for sub-directories, each containing data for one experiment. If using the STCS, the steps used to create the directory have been automated[1,17].

After the data has been stored in the directory, a program runs to convert the data files from MATLAB to Flexible Image Transport System (FITS) format[19,20]. The metadata stored in the FITS header is then uploaded into a database created in Research System, Inc. Interactive Data Language® (IDL). This database is searchable via a GUI interface or directly from the IDL command prompt[21].

When the data disk begins to fill up, the older data is archived onto CD-Rs. The IDL database tracks the file locations in order to maintain knowledge of the file locations. Three copies are made of each CD-R, one at JPL and one at GSFC are stored in a team member's office for general use. An additional reserve copy is stored in the laboratory in case something were to happen to the other two.

In the future, we hope to extend this database to monitor documents and add new file types.

7. WREQ BUG TRACKING SOFTWARE

To support the development of the STCS, we have implemented the WREQ bug tracking software developed by a group at Duke University. This is a web based system accessible to the WCT team, but not the outside world. Users can submit problems to the system via e-mail or an online form. The WCT software team will then act on the requests and update the web site with either the current status of the request. The software tracks resolved issues and includes features for monitoring the number of requests and the solution time. In addition to providing the user with a formal tool for incident reporting, this allows the programmers to have a central location to monitor and track software issues. [22]

Currently, this system is only in use for issues related to the STCS. In the future we would like to extend the capability to include development of the control software. Additionally, we would like to use similar software for experiment planning and scheduling.

8. SUMMARY

The laboratory computing environment supporting the WCT has evolved into a complex system with several different layers, developed by many different people. Without these layers, operation of the testbed would require multiple people physically located in the lab in order to perform even the simplest of tasks. The time taken to create the current system has been repaid many times over in the improved efficiency of the entire testbed operation.

There are many areas for improvement. As the testbed evolves, we plan to implement standards for control software, thus simplifying system maintenance. Additionally, we will evaluate the necessity of upgrading and replacing portions of the software.

ACKNOWLEDGEMENTS

The authors thank all those who have managed this system and seen this system evolve over the years. Special thanks to the WCT hardware designers and algorithm developers, without whom this system would be useless. This work was performed for the Next Generation Space Telescope project at NASA Goddard Space Flight Center and the Jet Propulsion Laboratory, California Institute of Technology.

The authors would like to specially thank: C. Bowers, D. Cohen, A. Chu, P. Davila, B. Dean, P. Dogoda, L. Feinberg, M. Fitzmaurice, J. Green, K. Ha, S. Hammers, K. Harvel, B. Hayden, C. LeBeouf, G. Mosier, T. Norton, D. Redding, F. Shi, D. Skelton, D. VanBuren, and B. Zukowski.

REFERENCES

1. S. Basinger, L. Burns, et al, "Wavefront sensing and control software for a segmented space telescope," SPIE Paper 4850-56, Waikoloa, Hawaii (2002).
2. P. Petrone, B. Zukowski, et al, "Optical design and performance of the NGST wavefront control testbed," SPIE Paper 4850-55 Waikoloa, Hawaii (2002).
3. F. Shi, A. Lowman, et al, "Segmented mirror coarse phasing with a dispersed fringe sensor: experiments on NGST's wavefront control testbed," SPIE Paper 4850-51, Waikoloa, Hawaii (2002).
4. D. Redding, C Bowers, et al, "Image-based wavefront sensing and control experiments," SPIE Paper 4850-62, Waikoloa, Hawaii (2002).
5. D. Cohen, D. Redding, "NGST high dynamic range unwrapped phase estimation," SPIE Paper 4850-52, Waikoloa, Hawaii (2002).
6. C. Ohara, D. Redding, et al, "PSF monitoring and in-focus wavefront control for NGST," SPIE Paper 4850-64, Waikoloa, Hawaii (2002).
7. F. Shi, C. Ohara, et al, "Segmented mirror coarse phasing with white light interferometry: modeling and experimenting on NGST's wavefront control testbed," SPIE Paper 4850-59, Waikoloa, Hawaii (2002).
8. A. Lowman, "Nexus Wavefront Control Testbed (WCT) Executive to Optics Control Computer (OCC) Interface Control Document (ICD), Revision 4," WCT Team Document, November 2000.
9. A. Lowman, D. Redding, et al, "Phase retrieval camera for testing NGST optics," ," SPIE Paper 4850-50, Waikoloa, Hawaii (2002).
10. J. Faust, A. Lowman, et al, "NGST phase retrieval camera design and experiment details," SPIE Paper 4850-61, Waikoloa, Hawaii (2002).
11. L. Wheeler, "Wavefront Control Testbed (WCT) Deformable Mirror (DM) Controller Design Document," WCT Team Document, April 2000.
12. L. Wheeler, "Wavefront Control Testbed (WCT) Deformable Mirror (DM) Controller User Guide," WCT Team Document, April 2000
13. L. Wheeler, "DCATT Executive to Deformable Mirror (DM) Controller Interface Document, Revision 3," WCT Team Document, July 1999.
14. L. Wheeler, J. Deering, "Wavefront Control Testbed (WCT) Accessory Controller (ACC) User Guide," WCT Team Document, April 2000.

15. K. Ha, D. Tonnu, “ACC FSM Command Structure”, WCT Team Internal Memo, June 2002.
16. L. Wheeler, J. Deering, “Wavefront Control Testbed (WCT) Monitor User Guide,” WCT Team Document, April 2000.
17. S. Stoner, L. Burns, et al. “Wavefront Control Testbed (WCT) Executive Software User’s Guide,” WCT Team Document, July 2002.
18. T. Norton, J. Schott, “Safety / Environmental System Cheat Sheet”, WCT Team Internal Memo, April 2000
19. NASA/GSFC Astrophysics Data Facility, “A User’s Guide for the Flexible Image Transport System, Version 4.0”, April 14, 1997
20. J. D. Ponz, et al, “The FITS image extension”, Astronomy & Astrophysics Supplement Series, 105, 53-55, 1994.
21. D. Lindler, “DCATT data archive operation instructions”, WCT Team Internal Memo, September 2001.
22. WREQ web site: http://www.math.duke.edu/~yu/wreq/